Future Computing and Informatics Journal

Volume 2
Issue 1 (2020) *2017, Volume 2, Issue 1*

Article 4

2017

Designing fuzzy rule base using Spider Monkey Optimization Algorithm in cooperative framework

Joydip Dhar
Department of Applied Sciences, ABV-Indian Institute of Technology & Management Gwalior,
jdhar.iiitmg@gmail.com

Available online at www.sciencedirect.com

ScienceDirect

Future Computing and Informatics Journal 2 (2017) 31–38
http://www.journals.elsevier.com/future-computing-and-informatics-journal/

Designing fuzzy rule base using Spider Monkey Optimization Algorithm in cooperative framework

Joydip Dhar [a,*], Surbhi Arora [b]

[a] Department of Applied Sciences, ABV-Indian Institute of Technology & Management Gwalior, Morena Link Road, Gwalior, 474015, India
[b] ABV-Indian Institute of Technology & Management Gwalior, Morena Link Road, Gwalior, 474015, India

Received 9 August 2016; revised 1 March 2017; accepted 26 April 2017
Available online 24 May 2017

Abstract

The paper focusses on the implementation of cooperative Spider Monkey Optimization Algorithm (SMO) to design and optimize the fuzzy rule base. Spider Monkey Optimization Algorithm is a fission-fusion based Swarm Intelligence algorithm. Cooperative Spider Monkey Algorithm is an off-line algorithm used to optimize all the free parameters in a fuzzy rule base. The Spider Monkeys are divided into various groups the solution from each group represents a fuzzy rule. These groups work in a cooperative way to design the whole fuzzy rule base. Simulation on fuzzy rules of two nonlinear controllers is done with a parametric study to verify the performance of the algorithm. It is observed that the root mean square error (RMSE) is least in the case of SMO than the other evolutionary algorithms applied in the literature to solve the problem of fuzzy rule designs like Particle Swarm Optimization (PSO), Ant Colony Optimization algorithm (ACO) algorithms.

Keywords: Evolutionary algorithm; SMO; Fuzzy system

1. Introduction

In the past two decades, many bio-inspired algorithms like genetic algorithms and Swarm Intelligence algorithms have been applied to optimize the fuzzy rule-based design [1–6]. Swarm Intelligence Algorithms such as Particle Swarm Intelligence Algorithms have proved themselves superior in optimizing the fuzzy rule base design. These algorithms draw inspiration from nature, i.e., they optimize the solutions by mimicking the behavior of animals such as birds, ants, etc. foraging for food.

Many advanced implementations of these algorithms have studied in the literature, one of them is swarm implementation that uses multiple swarms, optimizing different components of a solution vector and working in a cooperative framework to optimize the function. Different ideas to utilize these algorithms to solve the problem of fuzzy system design have been proposed in literature such as hierarchical cluster-based multi-species particle swarm optimization algorithm (HCMPSO) [1]. Again, cooperative continuous ant colony optimization algorithm (CCACO) [4], that uses ant colony optimization in continuous domain [5] and the other modifications of ACO [6], have proposed and analyzed by Refs. [7–9].

Recent years a Spider Monkey Optimization (SMO) algorithm in the cooperative framework is proposed [10]. It is a fission-fusion based algorithm that mimics the social behavior of spider monkeys as they forage for food [11]. The food searching of spider monkeys in groups is led by a female monkey. If the group leader is not able to find food for the group, it divides the group into multiple subgroups each lead by their own subgroup leader. These subgroups now forage independently for the food. Subgroups also communicate with each other to exchange information about food and territorial boundaries.

* Corresponding author.
E-mail address: jdhar.iiitmg@gmail.com (J. Dhar).
Peer review under responsibility of Faculty of Computers and Information Technology, Future University in Egypt.

http://dx.doi.org/10.1016/j.fcij.2017.04.004

Multiple groups of spider monkeys have been used to optimize the fuzzy rules with each group tuning a single fuzzy rule. Spider monkeys of all groups optimize a solution vector with respect to the free parameters of the fuzzy rule. All groups exchange the information of the local leader of the other groups to optimize their solution vector. The cooperative framework similar has been used in the literature to solve different problems [12–17].

Motivated from the SMO algorithm in cooperative framework [10], we develop an application of SMO in designing a fuzzy rule base. This paper is organized as follows: section 2 deals with the description of fuzzy systems to be optimized. Section 3 introduces spider monkey algorithm in a cooperative framework. The results and discussions on parametric settings through simulation are presented in section 4 and finally section 5 presents the conclusions.

2. Fuzzy control systems

The paper addresses the problem of accuracy-oriented fuzzy rule design. In Ref. [12] the researchers have described the mathematical model of the fuzzy system used in the problem, i.e., Tagaki-Sugeno-Kang (TSK) Fuzzy system. The rules in TSK fuzzy systems are:

$Rule_i$: If $x_1(k)$ is A_{i1} And .... And $x_n(k)$ is A_{in}, Then $u(k)$ is $f_i(x_1, , x_n)$.

Where $x_i(k)$ are the input variables, k is the time step, A_{ij} are the fuzzy sets defined by a Gaussian function as:

$$M_{ij} = exp\left\{ - \left(\frac{x_j - m_{ij}}{b_{ij}} \right)^2 \right\}, \tag{1}$$

where m_{ij} and b_{ij} represent the center and width of the fuzzy set A_{ij}, respectively.

The function in consequent part is defined as, the zero-order TSK systems:

$$f_i(x_1, \ldots, x_n) = a_{i0}, \tag{2}$$

and for first-order TSK systems:

$$f_i(x_1, \ldots, x_n) = a_{i0} + \sum_{j=1}^{n} a_{ij} x_j. \tag{3}$$

The number of rules defined in the fuzzy rule base gives the number of groups of Spider monkeys needed to optimize the problem. A fuzzy system with r given rules, r groups of spider monkeys would be applied to optimize the rules with each group, adjusting the free parameters m_{ij}, b_{ij}, and a_{ij} of the rule.

The solution vector of each Spider monkey optimizing a zero-order TSK system can be represented as:

$$\overrightarrow{s} = [m_{i1}, b_{i1}, \ldots, m_{in}, b_{in}, a_{i0}] \varepsilon\ \Re^{2n+1},$$

and the solution vector of the spider monkey optimizing a first-order TSK system is represented as:

$$\overrightarrow{s} = [m_{i1}, b_{i1}, \ldots, m_{in}, b_{in}, a_{i0}, .., a_{in}] \varepsilon\ \Re^{2n+1}.$$

In the inference engine, the fuzzy AND operation are implemented by the algebraic product in fuzzy theory. Thus, given an input data set $\overrightarrow{x} = (x_1, x_2 \ldots, x_n)$, the firing strength $\phi_i(\overrightarrow{x})$ of rule i is calculated by

$$\phi_i(\overrightarrow{x}) = \prod_{j=1}^{n} M_{ij}(x_j) = exp\left\{ - \sum_{j=1}^{n} \left(\frac{x_j - m_{ij}}{b_{ij}} \right) \right\}.$$

If there are r rules in a fuzzy system, the output of the system that is calculated by the weighted average defuzzification method:

$$u = \frac{\sum_{i=1}^{r} \phi_i(\overrightarrow{x}) f_i}{\sum_{i=1}^{r} \phi_i(\overrightarrow{x})}.$$

The next section describes the phases of SMO algorithm and its implementation in the cooperative framework to design the fuzzy rule system.

3. Spider Monkey Optimization Algorithm in cooperative framework

The SMO is based on fission-fusion characteristic of spider monkeys while they forage for their food. This study would map the two algorithms by applying SMO on the problem of accuracy-oriented fuzzy rule design that has already been solved using a modified version of ACO (CCACO) and do a parametric study on it. The study would further deal with modifications of SMO, i.e., steady state update of local and global leader, refinement in local leader decision phase and re-initialization of monkeys in each iteration to improve the accuracy of the optimizing algorithm.

3.1. Spider Monkey Optimization Algorithm

The spider monkey algorithm has been proved equally competitive in solving the problems of numerical optimization by the researchers [10,11]. The spider monkeys forage for food following a fission-fusion based social structure. The foraging behavior can be divided into four steps [10]:

1. The monkeys forage for food in a group of 40–50 members lead by a group leader (female monkey).
2. If the group leader is not able to find sufficient food for all the group members, she divides the group into subgroups consisting of 3–8 monkeys to reduce the competition. These groups now search for food independently.
3. Each subgroup is lead by a female monkey who is a local group leader. She is responsible for taking all the decisions for that subgroup.
4. Subgroups communicate with each other to exchange the information about availability of food and territorial boundaries.

In SMO algorithm, a group of spider monkeys is used to optimize the given function. Each spider monkey has its solution vector according to the fuzzy system designed as stated

in the previous section. Different phases of SMO are provided as follows:

3.1.1. Initialization of the population

The solution of each monkey is a D-dimensional vector where D is the number of parameters (variables) to be optimized. Each spider monkey corresponds to a potential settlement of the problem under consideration. The SMO initializes the spider monkeys with the uniformly distributed values between the maximum and minimum values.

$$SM_{ij} = SM_{minj} + U(0,1)*\left(SM_{maxj} - SM_{minj}\right), \tag{4}$$

where SM_{ij} is the j^{th} dimension of the i^{th} SM(Spidermonkey) and SM_{minj} and SM_{maxj} represent the maximum and minimum limits of that free parameter.

3.1.2. Local Leader Phase (LLP)

In this phase the spider monkeys (SM) update their values based upon the experience of the local leader of the subgroup and the monkeys in the same sub-group. The fitness of the new solution is calculated, if the fitness of the new solution is more than the original solution, the monkey updates its solution. The position update of i^{th} SM which is the part of k^{th} group is given as:

$$SM_{newij}=SM_{ij}+U(0,1)*\left(LL_{kj}-SM_{ij}\right)+U(-1,1)*\left(SM_{rj}-SM_{ij}\right). \tag{5}$$

where SM_{Ij} is the j^{th} dimension of the i^{th} SM, LL_{kj} represents the j^{th} dimension of the k^{th} local group leader position. SM_{rj} is the j^{th} dimension of the r^{th} SM which is chosen randomly within k^{th} group such that $r! = i, U(0,1)$ is a uniformly distributed random number between 0 and 1.

3.1.3. Global Leader Phase (GLP)

After completing local leader phase, the spider monkeys update their positions based on the experience of the global leader and the experience of the local group member. This is done in order to achieve better convergence.

$$SM_{newij}=SM_{ij}+U(0,1)*\left(GL_{j}-SM_{ij}\right)+U(-1,1)*\left(SM_{rj}-SM_{ij}\right), \tag{6}$$

where j is a number between [1,D], chosen randomly and GL is the Global Leader. The probability of monkeys to be selected for updating in this phase is proportional to its fitness value so that the better monkey has more chance of being selected and improving itself:

$$Prob_i = 0.9*\frac{fitness_i}{max\_fitness} + 0.1. \tag{7}$$

3.1.4. Global Leader Learning phase (GLL)

The position of global leader is updated by searching all the solutions. The position monkey with the highest fitness function is then taken as the position of the global leader. Also, it is checked that the position of the global leader has changed or not. If the position does not change the value of *GlobalLimitCount* is incremented by one.

3.1.5. Local Leader Learning phase (LLL)

In this phase the greedy search is done within the subgroups. The monkey with the highest fitness function is now making the local leader. The old and updated position of local leader is also checked. If it has not changed, then the *LocalLimitCount* is incremented by one.

3.1.6. Local leader decision phase (LLD)

If the position of local leader has not been updated up to a predefined threshold the *LocalLeaderLimit* the positions of the members of that group are updated. The new positions are set such that monkeys are attracted towards the global leader and are repelled by the local leader in order to avoid the stagnation of the solution:

$$SM_{newij}=SM_{ij}+U(0,1)*\left(GL_{j}-SM_{ij}\right)+U(0,1)*\left(SM_{ij}-LL_{kj}\right). \tag{8}$$

3.1.7. Global Leader Decision (GLD) phase

If the value of the global leader has not changed since *GlobalLeaderLimit* times, the global leader divides the populations into more groups for the diversification of the search. If the number of groups becomes equal to MG, the max group count. The global leader joins all the monkeys in a single group. Hence, it is a fission-fusion process. The LLL phase takes place after this process.

3.2. SMO in cooperative framework

The paper proposes an implementation of SMO in a cooperative framework to solve the problem of fuzzy rule design. To improve the accuracy further the paper proposes two modifications, i.e., (i) steady state update of local and global leader and (ii) reinitialization of least performing monkeys to the original SMO.

3.2.1. Cooperative framework

The study would deal with modifications of SMO, i.e., the steady state update of local and global Leader, refinement in local leader decision phase and re-initialization of monkeys in each iteration to improve the accuracy of the optimizing algorithm. To optimize a fuzzy system with r rules, initially r groups are initialized and then the rest steps are performed according to the series and parallel framework are shown in Algorithm 1 and Algorithm 2 respectively.

In the series implementation of SMO, all the steps of the algorithm are executed one after another for each iteration and the execution of algorithm for different groups occur in series with each other i.e., for each iteration firstly all the steps for first group of monkeys are executed and then all the steps for next group are executed and so on.

```
gp=1;
while gp ⇐ r do
    group[gp] → Initialize();
    group[gp] → LocalLeaderLearning();
    group[gp] → GlobalLeaderLearning();
end
i=1;
while i ⇐ end_iter do
    gp=1;
    while gp ⇐ r do
        group[gp] → LocalLeaderPhase();
        group[gp] → GlobalLeaderPhase();
        group[gp] → LocalLeaderLearning();
        group[gp] → LocalLeaderDecision();
        group[gp] → GlobalLeaderDecision();
    end
end
```

Algorithm 1. Implementation of SMO algorithm in Series Framework.

In the parallel framework implementation of SMO, each step of algorithm is executed simultaneously for each group and then the execution of next step takes place i.e., for each iteration a step would be executed for all groups of monkeys and then the next step of algorithm will get executed, see Algorithm 2.

```
gp=1;
while gp ⇐ r do
    group[gp] → initialize();
    group[gp] → LocalLeaderLearning();
    group[gp] → GlobalLeaderLearning();
end
i=1;
while i ⇐ end_iter do
    gp=1;
    while gp ⇐ r do
        group[gp] → LocalLeaderPhase();
    end
    gp=1;
    while gp ⇐ r do
        group[gp] → GlobalLeaderPhase();
    end
    gp=1;
    while gp ⇐ r do
        group[gp] → LocalLeaderLearning();
    end
    gp=1;
    while gp ⇐ r do
        group[gp] → LocalLeaderDecision();
    end
    gp=1;
    while gp ⇐ r do
        group[gp] → GlobalLeaderDecision();
    end
end
```

Algorithm 2. Implementation of SMO algorithm in Parallel Framework.

3.2.2. Fitness calculation

The fitness of a monkey that is analogous to the performance of that monkey is calculated by considering the global leaders of the other groups. This means that if the FS has *r* rules, the fitness of j^{th} monkey in the i^{th} group is calculated by considering the solutions of the global leaders of the other groups as the accompanying $r-1$ rules of the FS. Initially, since the fitness of the monkey cannot be determined, the first monkey from each of the group is taken as global leader of that group.

3.3. Steady state updation of local and global leader

In local leader phase of standard SMO algorithm, *LocalLeader* is used to create the new solutions of that subgroup. Similarly, *GlobalLeader* is used to create all the N new solutions in *Global-Leader-Phase*.To improve the performance of SMO, the steady state update of Local and Global Leaders was done i.e., as soon as the new solution generated it is compared with the existing Local or Global Leaders, if it is found to be better that the existing Local or Global Leader, the solution becomes the corresponding Local or Global Leader. It is an elite preservation process, i.e., as soon as a monkey with better solution is generated it starts contributing to the generation of new solutions.

3.4. Refinement in local leader decision phase

The Local Leader Decision phase deals with randomly initializing solutions of a subgroup if the local leader of that sub-group does not change for specified number of iterations. This phase helps in exploring more solutions and avoid stagnation of the solution. The proposed refinement deals with calculating the distance of a monkey with respect to global leader and changing its value in the neighborhood of the global leader. The refinement is done with a very small probability on a monkey in order to promote both diversity and search around the global leader.

3.5. Re-initialization of monkeys

The study proposes a modification in SMO that is to rank all the monkeys from best to worst according to the fitness after each iteration and randomly re-initialize the $N/4$ least performing monkeys where N is the number of monkeys in a group. This adds flexibility to the solutions of monkeys at each iteration. This modification adds explorations in early stages and exploitations in later stages.

4. Results and discussion

In the proposed algorithm, the first phase named local leader phase is used to explore the search region as in this phase all the members of the groups update their positions with high perturbation in the dimensions. The perturbation is high for initial iterations and gradually reducing in later iterations. The second phase Global Leader phase promotes the exploitation as in this phase, better candidates get more chance to update and in position update process, only single randomly selected dimension is updated. The third and fourth phase namely Local Leader Learning phase and Global Leader Learning phase, are used to check that the search process is

not stagnated. In these two phases, it is checked that the local best and global best solutions are updating or not in a predefined number of trials. If not then the solution is considered stagnated. The fifth phase Local Leader Decision phase is used to avoid the stagnation or premature convergence of local solutions. In this phase, if the local best solution is not updated in a predefined number of trials (*Local Leader Limit*) then all the members of that group are re-initialized. In this phase, all the dimensions of the individuals are initialized either randomly or by using global best solution and local best solution. Further, the Global Leader Decision phase is used to avoid stagnation of the global best solution. In this phase if the global best solution is not updated within a predefined number of trials (*Global Leader Limit*) then the group is divided into smaller subgroups.

4.1. Experimental setup

The study initializes the parameters for the SMO in the cooperative framework as:

Number of rules: 5.
Number of test cases: 250.
Size of Group: 50.
Max number of subgroups: 4.
Global Leader Limit: 50.
Local Leader Limit: 1500.
Perturbation Rate: 0.4.

4.2. Simulation

Motivated from the nonlinear plant-tracking control problem [4], fuzzy rules in nonlinear plant-tracking control were optimized using SMO in cooperative framework with steady state updating of local and global leaders and re-initialization of least performing monkeys after each iteration.

In this problem, the plant to be controlled is described by:

$$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k), \tag{9}$$

where $-2 < y(k) < 2$, with $y(0) = 0$, $u(k)$ is the control input, and $u(k)\varepsilon[-1, 1]$. As in previous studies, the objective is to control the output $y(k)$ to track the following desired trajectory by an FS:

$$y_d(k) = \sin(\pi k/50)\cos(\pi k/30), \tag{10}$$

where, $1 <= k <= 250$. The designed FS inputs are $y_d(k+1)$ and $y(k)$, and the output is $u(k)$. As in a previous study [26], the error function E for performance evaluation is defined to be the root-mean-squared error (RMSE), i.e.,

$$E = \sqrt{\frac{\sum_{k=0}^{249}(y_d(k+1) - y(k+1))^2}{250}}. \tag{11}$$

The fitness function used in this case was $\frac{1}{(1+RMSE)}$, where RMSE is Root Mean Square Error. Fig. 1 shows the changes in the fitness values with the number of iterations. The graph shows that the fitness value obtained were almost similar, varying only in at the fourth decimal place. There is a dip after 1500 iterations in some cases signifying Local Leader Decision Phase, if the Local Leader doesn't change after a certain number of iterations denoted by *LocalLeaderLimit* (taken 1500 in our implementation), the solution vectors of monkeys are reinitialized. Table 1 shows the complete fuzzy rule set generated using SMO.

4.3. Parametric study

In the above study of SMO values of certain parameters such as group size (equal to 50), perturbation rate (0.4), maximum subgroups allowed (5) and swarm size (50) were kept based on some preliminary experiments. To fine tune (finding most suitable values) these parameters, sensitivity analysis with different values of these parameters have been carried out in this section.

First, we compared the value of pr (perturbation rate) i.e., the probability at Local Leader Decision Phase that a monkey will be randomly initialized a value. The perturbation rate varied from 0.1 to 0.9 keeping the other parameters fixed as defined above and a graph was plotted as shown in Fig. 2. The graph clearly depicts that the best fitness could be achieved when the value of pr is taken in between 0.6 and 0.7, i.e., the algorithm performs better when the turbulence factor is high.

Next, we compared the value of mg that denotes the maximum number subgroups that can be formed within a group. In the experiments the value of mg varied from 1 to 6 and the analysis was done for the best fitness, that could be achieved as shown in Fig. 3. The results depicted that the best fitness was achieved when the allowed mg was kept low at 3.

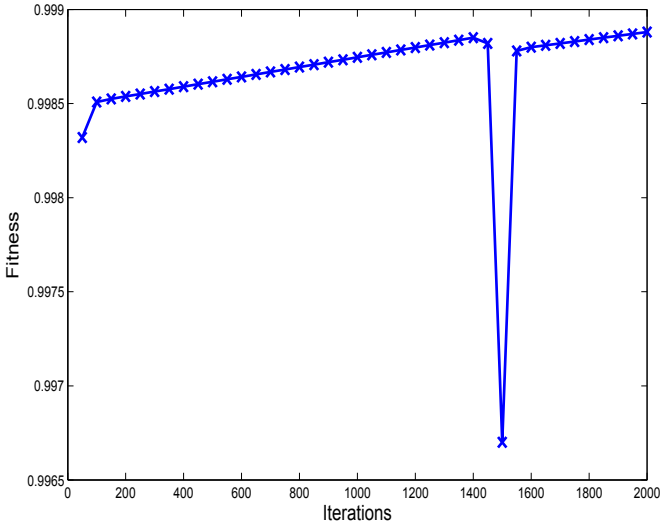

Fig. 1. Fitness of SMO with iterations.

Further, we compared the fitness with respect to the swarm size taken for optimizing by doing experiments with swarm size varying from 40 to 160 monkeys with the step size of 20 as compared the fitness obtained as shown in Fig. 4. The studies show that the result is sensitive towards the swarm size, it increases with the swarm size as there are more number of monkeys involved to optimize the rule-base till a certain point. The optimum value of swarm size can be observed at 80.

A comparison study on the root mean square error of proposed SMO with the existing evolutionary algorithms given in literature presented in Table 2. The table shows that the SMO performs marginally better than the other algorithms. The better performance of SMO is because it works on the phenomenon of dynamic fission and fusion which help in optimizing both the local and global solutions. Since initially there is a single group, so all the solutions optimized towards a single solution. But after a predefined number of iterations further optimization is not taking place, then the monkeys divide into multiple groups to ensure that all the options explored, and the result doesn't get stagnated over a local optimum. The global leader decision phase ensures that all the possible local optima visited, and the solution is optimized considering them, whereas local leader decision stage provides that the local solutions not stagnated.

Leader decision phase when a monkey will randomly be initialized value. The perturbation rate was varied from 0.1 to 0.9 keeping the other parameters fixed for different implementations of SMO as shown in Fig. 5. The graph clearly depicts that the best fitness could achieve at different *pr* value between 0.6 and 0.8 for different modifications of SMO.

Our studies have shown that the SMO evolutionary algorithms produce better results in a cooperative framework than when they are used to optimize the solution alone [1,11–17].

5. Conclusions

This paper proposes an application of a budding evolutionary algorithm, Spider Monkey Optimization Algorithm in a cooperative framework with two slight modifications, i.e., *Steady state update* and *reinitialization of least performing monkeys* to optimize the fuzzy rule bases. SMO has applied on a fuzzy system, and the results prove that the root means square error is least in the case of SMO than the other evolutionary algorithms applied in the literature to solve the problem of fuzzy rule designs like PSO, ACO algorithms. In multi-group cooperative framework, it can't be guaranteed that the solution obtained is a global optimum, and the algorithm has not got stagnated on a local optimum since the result is the combination of the partial solutions. Using SMO in the cooperative framework, we can ensure that partial solutions don't result in local optima better than other evolutionary algorithms due to its nature of breaking into groups dynamically. It is observed from Table 2 that SMO outperformed other evolutionary algorithms. Fig. 5 depicts that the best fitness could achieve at perturbation rate (*pr*) around 0.7 for different modifications of SMO.

The Spider Monkey Optimization Algorithm has proved itself to be equally competitive and even better algorithm than the other swarm intelligence algorithm. The beauty of this

Table 1
Designed Rule-Base for the problem.

Part	Antecedent				Consequent
Variables	$y(k)$		$y_d(k+1)$		$u(k)$
Parameters	m_{i1}	b_{i1}	m_{i2}	b_{i2}	a_{i0}
Rule1	0.651566	0.297857	−0.504386	0.699871	−0.88707
Rule2	−0.765195	0.281799	0.297417	0.304275	0.579535
Rule3	0.971301	0.67363	−0.701304	0.250517	−0.436353
Rule4	0.138874	0.683069	0.827576	0.364216	0.790475
Rule5	0.166164	0.459015	−1.51882	0.502406	−0.814491

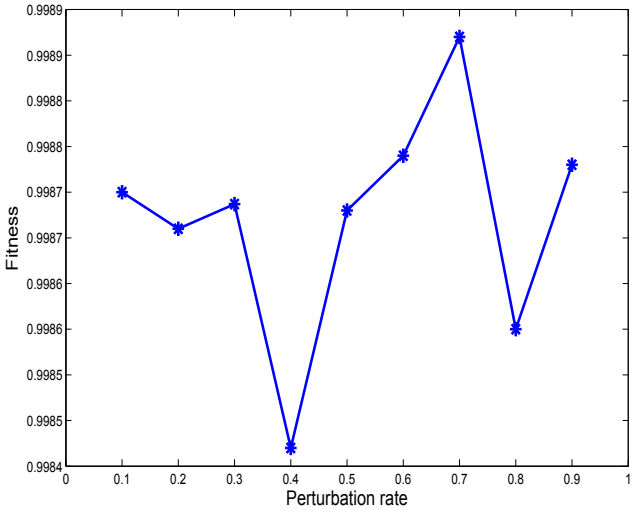

Fig. 2. Effect of perturbation rate on fitness of monkeys.

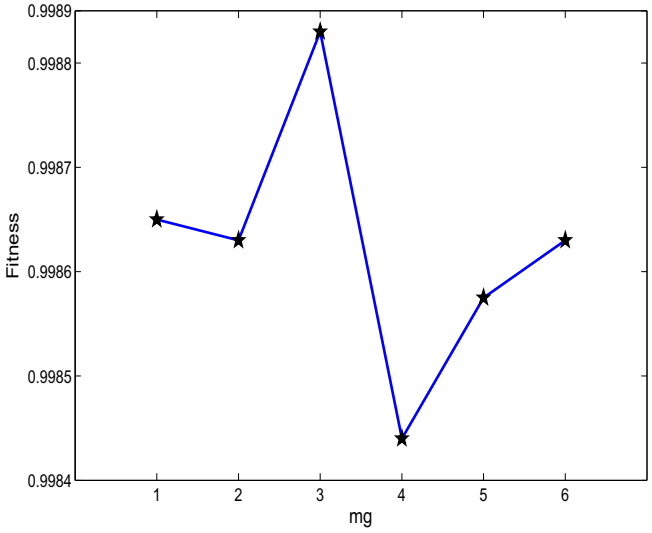

Fig. 3. Effect of the maximum allowed sub-groups on fitness of monkeys.

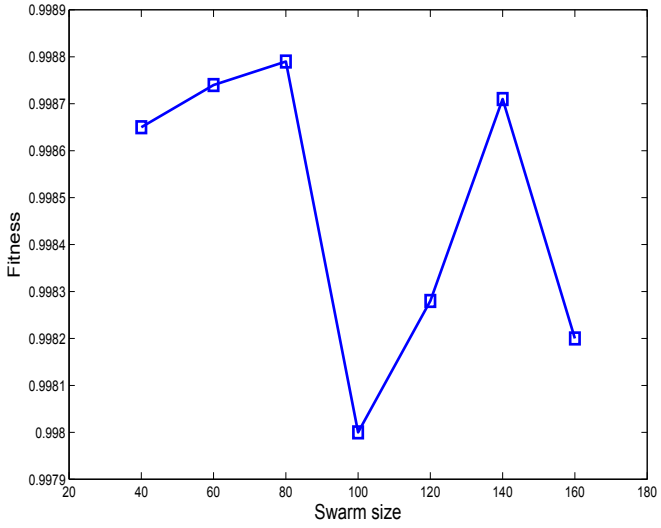

Fig. 4. Effect of swarm size on fitness of monkeys.

Table 2
RMSE Comparison for different Evolutionary Algorithms.

Algorithms	HCMSPSO [1]	CCACO [4]	Proposed SMO
RMSE	0.0389	0.0223	0.0119

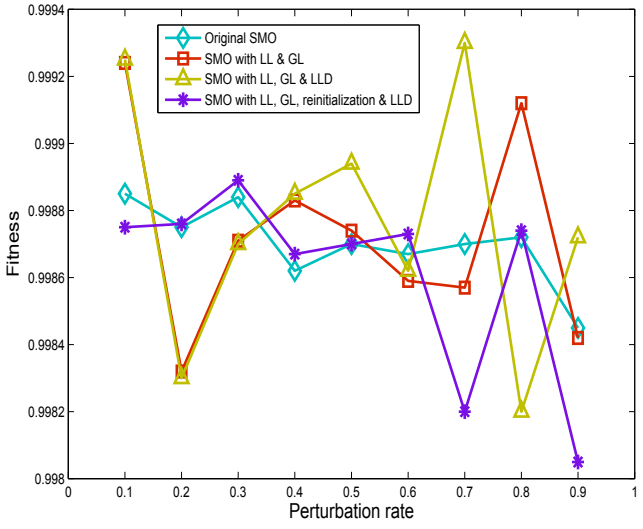

Fig. 5. Perturbation rate comparison for different implementations of SMO.

algorithm lies in the natural fusion of multiple monkey subgroups into one and fission of subgroups for searching the solution. The research focuses on modifying the original Spider Monkey Algorithm to make it suitable to be applied in different application domains. The algorithm is firstly applied in a cooperative framework for designing fuzzy rule base, and the parametric study is done on the various parameters to determine the optimal solution. For the improvement of the efficiency of the algorithm, different modifications proposed viz., Steady state update of Local and Global Leader, Refinement in Local Leader Decision Phase and re-initialization of monkeys in each iteration and the modified algorithms are applied to design fuzzy rule base. In future, SMO can be further applied to other Swarm Intelligence techniques to make the algorithm an online learning approach by even determining the number of rules necessary for the fuzzy system and then designing those rules.

The benefit of this structured group strategy is that initially there is a single group so every newly generated food source is attracted towards the best food source (in this case the global best will be the local best also), thereby converging faster to the solution. But as a result of such exploitative tendency, in many cases, the population may skip the global minima and can get stuck into local minima. Therefore, to avoid this situation, if global minima are not updating itself for a predefined number of times then the group is divided into subgroups. Now every new solution will be attracted towards the respective subgroups local best food source, hence contributes in the exploration of the search space. When the maximum number of subgroups have formed, this phase helps to balance the exploration and exploitation capability of the algorithm while maintaining the convergence speed.

References

[1] Juang C-F, Hsiao C-M, Hsu C-H. Hierarchical cluster-based multispecies particle-swarm optimization for fuzzy-system optimization. Fuzzy Syst IEEE Trans 2010;18(1):14–26.
[2] Sharma H, Bansal J, Arya K. Self balanced differential evolution. J Comput Sci 2014;5:312–23.
[3] Bansal J, Singh P, Deep K, Nagar A, Pant M. Special issue on bio-inspired computing: theories and applications. J Comput Sci 2014;5:135–6.
[4] Juang C-F, Hung C-W, Hsu C-H. Rule-based cooperative continuous ant colony optimization to improve the accuracy of fuzzy system design. Fuzzy Syst IEEE Trans 2014;22(4):723–35.
[5] Socha K, Dorigo M. Ant colony optimization for continuous domains. Eur J Oper Res 2008;185(3):1155–73.
[6] Dorigo M, Birattari M. Ant colony optimization. In: Encyclopedia of machine learning. Springer; 2010. p. 36–9.
[7] Juang C-F, Chang P-H. Designing fuzzy-rule-based systems using continuous ant-colony optimization. Fuzzy Syst IEEE Trans 2010;18(1):138–49.
[8] Juang C-F, Lu C-M, Lo C, Wang C-Y. Ant colony optimization algorithm for fuzzy controller design and its fpga implementation. Ind Electron IEEE Trans 2008;55(3):1453–62.
[9] Juang C-F, Hsu C-H. Reinforcement interval type-2 fuzzy controller design by online rule generation and q-value-aided ant colony optimization, Systems, Man, and Cybernetics, Part B: Cybernetics. IEEE Trans 2009;39(6):1528–42.
[10] Bansal JC, Sharma H, Jadon SS, Clerc M. Spider Monkey Optimization Algorithm for numerical optimization. Memetic Comput 2014;6(1):31–47.
[11] Kumar S, Sharma VK, Kumari R. Modified position update in Spider Monkey Optimization Algorithm. Int J Emerg Technol Comput Appl Sci (IJETCAS) 2014;7(2):198–204.
[12] Potter MA, De Jong KA. A cooperative coevolutionary approach to function optimization. In: Parallel problem solving from naturePPSN III. Springer; 1994. p. 249–57.
[13] Van den Bergh F, Engelbrecht AP. A cooperative approach to particle swarm optimization. Evol Comput IEEE Trans 2004;8(3):225–39.

[14] Yu Y, Xinjie Y. Cooperative coevolutionary genetic algorithm for digital iir filter design. Ind Electron IEEE Trans 2007;54(3):1311−8.
[15] Yang Z, Tang K, Yao X. Large scale evolutionary optimization using cooperative coevolution. Inf Sci 2008;178(15):2985−99.
[16] Lin C-J, Chen C-H, Lin C-T. A hybrid of cooperative particle swarm optimization and cultural algorithm for neural fuzzy networks and its prediction applications, Systems, Man, and Cybernetics, Part C: applications and Reviews. IEEE Trans 2009;39(1):55−68.
[17] Li X, Yao X. Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms. In: Evolutionary computation, 2009. CEC'09. IEEE Congress on, IEEE; 2009. p. 1546−53.